



[Previous Doc](#)      [Next Doc](#)      [Go to Doc#](#)

## Freeform Search

**Database:** 
 US Pre-Grant Publication Full-Text Database  
 US Patents Full-Text Database  
 US OCR Full-Text Database  
 EPO Abstracts Database  
 JPO Abstracts Database  
 Derwent World Patents Index  
 IBM Technical Disclosure Bulletins

**Term:** L4 AND AUXILIARY

**Display:** 50 **Documents in Display Format:** FRO **Starting with Number** 1

**Generate:** ☐ Hit List ☒ Hit Count ☐ Side by Side ☐ Image

Search

Clear

Interrupt

### Search History

**DATE:** Wednesday, October 04, 2006

[Purge Queries](#)

[Printable Copy](#)

[Create Case](#)

#### Set Name Query

side by side

#### Hit Count Set Name

result set

DB=USPT; PLUR=YES; OP=OR

<u>L7</u>	L5 AND (REMOV\$ NEAR AUXILIARY)	1	<u>L7</u>
<u>L6</u>	L5 AND (REMOV\$ NEAR AUXILIARY)	0	<u>L6</u>
<u>L5</u>	L4 AND AUXILIARY	4	<u>L5</u>
<u>L4</u>	L3 AND (FIRST NEAR ENTRY)	43	<u>L4</u>
<u>L3</u>	IDENTITY NEAR SYSTEM	894	<u>L3</u>
<u>L2</u>	L1 AND (FIRST NEAR ENTRY)	116	<u>L2</u>
<u>L1</u>	ENTITY NEAR SYSTEM	1763	<u>L1</u>

END OF SEARCH HISTORY



Go to Doc#

Print

Jan 6, 2004

TITLE: Request based caching of data store data

Identity Systems have become more popular with the growth of the Internet and the use of networks and other information technologies. In general, an Identity System provides for the creation, removal, editing and other management of identity information stored in various types of data stores. The identity information pertains to users, groups, organizations and/or things. For each entry in the data store, a set of attributes is stored. For example, the attributes stored for a user may include a name, address, employee number, telephone number, email address, user ID and password. The Identity System can also manage access privileges that govern the subject matter an entity can view, create, modify or use in the Identity System.

Identity System users direct the operation of the Identity System by submitting requests that call for an Identity System response, such a searching and viewing a user's profile. Requests frequently require the Identity System to repeatedly access the same entries in the Identity System's data store. For example, a request may cause the Identity System to load data into a data store entry and later retrieve the newly loaded data multiple times for performing different functions. This can occur when a client provides identification information that is stored in a data store entry. The request may retrieve this information on multiple occasions for forwarding to servers or applications accessed by the client request.

Each client request is assigned to a thread of execution in an Identity Server within an Identity System. A cache object is associated with the thread of execution for caching data store entry accesses arising from the request. In one implementation, the thread of execution contains a thread local storage with a pointer to the cache object. Employing the cache object to maintain frequently accessed data store entries reduces the number of data store accesses required to service the request--speeding request processing time and freeing data store bandwidth.

In one embodiment of the present invention, an Identity Server receiving a request interprets a command in the request to call for an access to a data store entry. In response to the command, the Identity Server accesses a first entry in the cache object that corresponds to the data store entry called for in the command. If the cache object does not include an entry corresponding to the data store entry, the Identity Server creates and loads a corresponding entry in the cache object.

In further embodiments of the present invention, request based caching is employed in processing environments other than an Identity System. Request based caching can have broad applicability to enhance the performance of many different server based systems.

FIG. 6 is a flow chart describing one embodiment of a process for accessing the Identity System.

Drawing Description Text (33):

Drawing Description Text (34):

Drawing Description Text (35):

Detailed Description Text (2):

Detailed Description Text (3):

Detailed Description Text (11):

Detailed Description Text (15):

Detailed Description Text (18):

The third application in the Identity System, Organization Manager 46, streamlines the management of large numbers of organizations within an e-business network, including partners, suppliers, or even major internal organizations such as sales offices and business units. Certain infrastructure security and management operations are best handled--or can only be handled--at the highest organizational unit level rather than at the individual or group level. Like User Manager and Group Manager, this application relies on multi-step workflow and

delegation capabilities. Organization Manager handles the following administrative tasks: (1) organization lifecycle management, whereby companies can create, register, and delete organizations in their systems using customizable workflows; (2) maintenance of organization profiles on an attribute-by-attribute basis through self-service, delegated administration and system-initiated activities; (3) organization self-registration, whereby organizations such as business partners, customers and suppliers can self-generate a request to be added to the e-business network; and (4) creation of reusable rules and processes through multi-step workflows.

Detailed Description Text (20):

With the system of FIG. 1 deployed, Web Server 18 (enabled by Web Gate 28, Access Server 34, and Directory Server 36) can make informed decisions based on default and/or specific rules about whether to return requested resources to an end user. The rules are evaluated based on the end user's identity profile, which is managed by the Identity System. In one embodiment of the present invention, the general method proceeds as follows. An end user enters a URL or an identification of a requested resource residing in a protected policy domain. The user's browser sends the URL as part of an HTTP request to Web Server 18. Web Gate 28 intercepts the request. If the end user has not already been authenticated, Web Gate 28 causes Web Server 18 to issue a challenge to the browser for log-on information. The received log-on information is then passed back to Web Server 18 and on to Web Gate 28.

Detailed Description Text (37):

Database proxy 154 encapsulates the supporting agent objects for the particular operation. It also acts as a storage area where input parameters and output results are stored. Each database proxy object exposes its methods and input parameters. These parameters include search base, object class, auxiliary class, filter, search scope, attributes and entry. After a database client sets all the parameters, the client calls the execute method of the proxy to invoke the database operation. The client then calls the database proxy GetResults method to retrieve the operations results.

Detailed Description Text (50):

FIG. 5 shows a hierarchical tree. Some organizations employ fat or flat trees for ease of maintenance. A flat directory tree is a directory information tree that does not have any hierarchy. All of the nodes are leaf nodes (nodes without any child nodes). A fat directory tree is a tree that has a large number of nodes at any given level in a directory information tree. One advantage of a fat or flat tree is user maintenance. For example, if an employee moves to a new group, the node must be moved to a new container if the tree is not flat or fat. By moving the node to a new container, the distinguished name for the node changes and all certificates become void. One drawback of flat or fat trees is that the organization loses the benefits of having a logical directory, such as using the logical directory to determine who has access to which nodes. To remedy this, the Identity System includes partition support for fat and flat tree directories using filters. From a configuration page, an attribute can be configured to be accessible (read, modify, etc.,) based on a two part filter. The first component in the filter identifies a top node in the directory. The filter will only apply to those entities at or below that top node. The second component of the filter is an LDAP filter which defines who can access the attribute. This two component filter can be applied on an attribute by attribute basis.

Detailed Description Text (51):

There are many ways for an entity to access and use the Identity System. In one embodiment, the entity can access the Identity Systems services using a browser. In other embodiments, XML documents and API's can be used to access the services of the Identity System. For example, an entity can use a browser by pointing the browser to Identity Server 40. The user will then be provided with a login page to enter the user's ID, password, type of user and application requested (optional). Upon filling out that information, the user will be authenticated and authorized (by the Access System) to use the Identity System, as described below. Alternatively, the Access System can be bypassed (or there may be no Access System) and the Identity System authenticates the user.

Detailed Description Text (52):

Detailed Description Text (53):

Detailed Description Text (54) :

Detailed Description Text (55):

Detailed Description Text (56):

http://jupiter:9000/bin/cgi-bin/accum\_query.pl?MODE=%20%20%20%20Display%20%20%20%20... 10/4/06



Detailed Description Text (71):

As discussed above, when an entity logs into the Identity System, the entity indicates the entity's role. There are at least six roles: System Administrator, Master Identity Administrator, Master Access Administrator, Delegated Access Administrator, Delegated Identity Administrator and End User. The System Administrator can perform all Access System configuration tasks and all Identity System configuration tasks. The Master Identity Administrator can configure access controls, attribute access controls, new user services, workflow definitions, setting the search base, delegating rights, expanding dynamic groups, and setting container limits. The Master Access Administrator can configure a web gate, configure an access server, create host identifiers, configure users, set-up policies and policy domains, and delegate rights. The Delegated Identity Administrator is an administrator who has been delegated rights from the Master Identity Administrator. The Delegated Access Administrator can be delegated rights from a Master Access Administrator. An End User cannot perform configuration functions. There can also be a delegated admin who can create/delete users, add/remove users to/from groups, process workflow steps, etc.

Detailed Description Text (73):

One right that an administrator has and which can be delegated to a Delegated Administrator is the proxy right. The proxy right for person A allows person A to choose another person (e.g. person B) to be a proxy for person A during a period of time. For example, if a Delegated Administrator (or other administrator) is going on vacation, or will otherwise be unavailable to perform its administrative duties, that Delegated Administrator can identify another person (or persons) who can be a proxy for that Delegated Administrator. While person B is being a proxy for person A, person B has all the rights and privileges of person A within the Identity System. Person B does not have the rights of person A in the Access System. Thus, the Identity System will see person B as person A, but the Access System will see person B as person B.

Detailed Description Text (77):

In step 668, the UidCookie on the user's machine is edited by changing Uid 362 to equal the user identification for the person being proxied. In step 670, the user now operates as the person being proxied in the Identity System. Because the Uid in the Cookie identifies the person being proxied, the Identity System treats the user as the person being proxied. However, the UidCookie is only used by the Identity System, so only the Identity System treats the person as the person being proxied. The Access System uses a different cookie (described below), and the Access System's cookie is not edited. Therefore, the Access System treats the user as himself or herself and not as the person being proxied. While being a proxy, the user has all the rights and privileges as the person being proxied. In one embodiment, the process of FIG. 15 is performed without the user providing or knowing the password for the person being proxied and, therefore, without authenticating the password and ID for the person being proxied.

Detailed Description Text (78):

In one embodiment, step 670 includes receiving a request from the user (e.g. the entity who is the proxy) to access a service of the Identity System. In response, the system will access the Uid in the cookie, and use that Uid to access attributes, group memberships and organizations memberships for the identity profile of the person being proxied. Based on those attributes, the user will or will not be provided access to the requested service.

Detailed Description Text (80):

A lot of the tasks that are performed in the Identity System are accomplished using workflows. A workflow is a predefined set of steps that perform a specific task, where information or tasks are passed between participants and programs according to a defined set of rules. One embodiment of the present invention supports the following types of workflows: create object; delete object; change the value of attributes; and certificate issuance, revocation and renewal. In one embodiment of the present invention, a user is required to create a workflow to create or delete an object, change the value of an attribute or implement certificates. Workflows ensure that an organization's guidelines for performing a task are met. Workflows can be defined in the User Manager, Group Manager or Organization Manager. A workflow can be used only in the application (e.g. User Manager) in which it was created. Each workflow has two or more steps, including one to start the action and one to implement or commit it. Each step can

Detailed Description Text (96):

Detailed Description Text (108) :

Detailed Description Text (112):

Detailed Description Text (126):

Detailed Description Text (147) :

http://jupiter:9000/bin/cgi-bin/accum\_query.pl?MODE=%20%20%20%20Display%20%20%20%20... 10/4/06

Detailed Description Text (148):

Detailed Description Text (149) :

Detailed Description Text (150) :

Detailed Description Text (151):

http://jupiter:9000/bin/cgi-bin/accum\_query.pl?MODE=%20%20%20%20Display%20%20%20%2... 10/4/06

Detailed Description Text (152):

Detailed Description Text (153):

Detailed Description Text (154):

Detailed Description Text (174):

http://jupiter:9000/bin/cgi-bin/accum\_query.pl?MODE=%20%20%20%20Display%20%20%20%20%20... 10/4/06

In one embodiment, Identity Server 40 obtains attribute display characteristics from directory entries in Directory Server 36. Each Directory Server entry corresponds to a different attribute type. For each attribute, Identity Server 40 locates a corresponding directory entry, which provides the attribute's display characteristics. In one such embodiment, a system administrator creates all the display attribute directory entries when Identity System 40 is configured. In alternate embodiments of the present invention, the directory entries are replaced by tables, data structures, or other means that relate display characteristics to attributes so the display characteristics can be obtained by Identity Server 40.

Requests for data received by the Identity System frequently require repeated access to the same entries in Directory Server 36. Continually retrieving this information through Directory Server 36 slows operation and wastes server bandwidth. Therefore, Identity Server 40 provides each active request with a cache to reduce the number of data store accesses.

As described above, clients submit requests to the Identity System asking for information on requesting tasks to be performed. These requests can be submitted via HTTP, XML documents, or other means. In some embodiments of the present invention, multiple Identity Servers are employed to increase the throughput of the Identity System. In such embodiments, requests are assigned to Identity Servers so as to balance the load of each Identity Server. In some instances a request may execute a function that requires a primary Identity Server handling the request to communicate with another Identity Server.

After executing the local operation or if no local operation is required, management service 1910 opens a message channel for providing the remote request to remote Identity Server 1902 (step 1966). Management service 1910 then issues the remote request to remote Identity Server 1902 (step 1968). In the embodiment shown in FIG. 46, management service 1910 opens up a communication channel with Identity Server 1902 and provides the remote request to server 1902. In alternate embodiments, however, more than two Identity Servers are employed in the Identity System. In such embodiments, local Identity Server 1900 opens message channels with all the other remote Identity Servers and issues the remote request to them.

Embodiments of the present invention provide for establishing different sets of criteria for obtaining a certificate. For example, a high level person in an organization may have great need for access to confidential corporate information. The corporation may wish to issue this person a certificate without any more than a mere request being filed. On the other hand, entry level employees at a corporation may have very little need for access to confidential information. The corporation may wish to have the entry level person's manager approve the issuance of a certificate. One embodiment of the integrated Access and Identity System of the present invention incorporates certificate management into the workflow process so different standards for certificate management can be applied among various entities. In one implementation, different certificate enrollment, renewal, and revocation workflows can be defined for different types of system users.

The integrated Access and Identity System of the present invention also includes Certificate Processing Server 2076, which is in communication with Identity Server 40 to communicate with certificate registration module 2072. Certificate Processing Server 2076 issues certificate signing requests to Certificate Authority 2084, which is external to the integrated Access and Identity System and in communication with Certificate Processing Server 2076. Certificate Authority 2084 is typically a third party vendor that provides certificates, including pairs of public and private keys for attachment to the certificates. One example of a third party certificate provider is Verisign. Certificate Processing Server 2076 is also in communication with signing device 2078. Signing device 2078 digitally signs certificate signing requests before they are issued to Certificate Authority 2084. Digitally signing certificate signing requests heightens the level of security in the connection between Certificate Processing

Detailed Description Text (232):

Detailed Description Text (237):

Detailed Description Text (253) :

Detailed Description Text (257):

Detailed Description Text (258):

Detailed Description Text (259):

http://jupiter:9000/bin/cgi-bin/accum\_query.pl?MODE=%20%20%20%20Display%20%20%20%2... 10/4/06

Detailed Description Text (260):

Detailed Description Text (261):

Detailed Description Text (262):

Detailed Description Text (265):

Detailed Description Text (292):

CLAIMS: